**Requirements - Multi cloud support for object versioning**

This document describes the requirements and use cases for support for versioned object storage in SODA. These have been arrived at after looking primarily at the Versioning support in AWS S3, Azure (or the lack of it here) and GCP object storage.

This document also lists the cross feature interactions with versioning and

1. Object migration in SODA
2. Object lifecycle in SODA

Requirements

1. Support versioning on a bucket
2. Support suspend versioning on a bucket
3. Support resume versioning on a bucket
4. Support the below state transitions on a bucket w.r.t versioning



1. Support recovery from accidental deletion or overwrite. For example:
   1. If you delete an object, you can always restore the previous version
   2. If you overwrite an object, it results in a new object version in the bucket. You can always restore the previous version
2. For a versioning enabled bucket, support
   1. Adding Objects
      1. Using all supported means, including but not limited to the UI, API and CLI
   2. Listing Objects
      1. Using all supported means, including but not limited to the UI, API and CLI
      2. Each request returns up to 1,000 versions, unless you specify a lower number. If the bucket contains more versions than this limit, you send a series of requests to retrieve the list of all versions. This process of returning results in "pages" is called pagination. To show how pagination works, the examples limit each response to two object versions. After retrieving the first page of results, each example checks to determine whether the version list was truncated. If it was, the example continues retrieving pages until all versions have been retrieved
      3. As an example, In your bucket, if you have 2 objects a and b, a has 900 versions and b has 300 versions, the list call will return 900 versions of a and the most recent 100 versions of b (total 1000)
      4. Object versions will be returned in the order in which they were stored, with the most recently stored returned first
      5. Retrieval of a subset of all object versions in a bucket, for example, retrieve all versions of a specific object (filter by name or object identifier)
      6. When the response exceeds the max-value (1000), a second request can be submitted to retrieve the remaining object versions
   3. Retrieving Object Versions
      1. Using all supported means, including but not limited to the UI, API and CLI
      2. By default, always retrieve the most recent version
      3. Retrieval of a specific version of an object by specifying the Version Id
   4. Deleting Object Versions
      1. Using all supported means, including but not limited to the UI, API and CLI
      2. On a versioning enabled bucket, DELETE will not delete the object permanently
      3. To delete the object permanently, support ‘DELETE Object VersionId’
   5. Transitioning Object Versions
      1. Using all supported means, including but not limited to the UI, API and CLI
      2. An *Expiration* action that applies to the current object version and instead of deleting the current object version, marks it as non-current
      3. A *NoncurrentVersionExpiration* action that applies to noncurrent object versions, which will be permanently removed and cannot be recovered
   6. Restoring Previous Versions
      1. Using all supported means, including but not limited to the UI, API and CLI
      2. To Copy a previous version of the object into the same bucket. The copied object becomes the current version of that object and all object versions are preserved
      3. To Permanently delete the current version of the object. This, in effect, turns the previous version into the current version of that object
   7. Versioned Object Permissions
      1. PUT Object ACL Version Id to set the ACL for a specific object version
      2. GET Object Version Id to get the ACL for a specific object version
3. For a versioning suspended bucket, support
   1. Adding objects
   2. Getting objects
   3. Deleting objects
4. All the above requirements (except this one) are to be supported for the following Object stores
   1. AWS S3 <https://aws.amazon.com/s3/>
   2. GCP <https://cloud.google.com/storage/docs/json_api/v1/objects>
   3. Azure Blob storage <https://docs.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction>
   4. HWS <https://intl.huaweicloud.com/en-us/product/obs.html>
   5. CEPH <https://docs.ceph.com/docs/mimic/radosgw/>
   6. YIG <https://github.com/journeymidnight/yig>